LLNL-CONF-424662

Enhanced Named Entity Extraction via Error-Driven Aggregation

T. D. Lemmond, N. C. Perry, J. W. Guensche, J. J. Nitao, R. E. Glaser, P. Kidwell, W. G. Hanley

February 26, 2010

International Conference on Data Mining 2010
Las Vegas, NV, United States
July 12, 2010 through July 15, 2010

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

Enhanced Named Entity Extraction via Error-Driven Aggregation

T. Lemmond[1], N. Perry[2], J. Guensche[1], J. Nitao[1], R. Glaser[1], P. Kidwell[1], and W. Hanley[1]
[1]Lawrence Livermore National Laboratory, Livermore, CA, USA
[2]Mathematics Department, Brigham Young University, Provo, UT, USA

Abstract - *Despite recent advances in named entity extraction technologies, state-of-the-art extraction tools achieve insufficient accuracy rates for practical use in many operational settings. However, they are not generally prone to the same types of error, suggesting that substantial improvements may be achieved via appropriate combinations of existing tools, provided their behavior can be accurately characterized and quantified. In this paper, we present an inference methodology for the aggregation of named entity extraction technologies that is founded upon a black-box analysis of their respective error processes. This method has been shown to produce statistically significant improvements in extraction relative to standard performance metrics and to mitigate the weak performance of entity extractors operating under suboptimal conditions. Moreover, this approach provides a framework for quantifying uncertainty and has demonstrated the ability to reconstruct the truth when majority voting fails.*

Keywords: Knowledge discovery, text mining, named entity extraction, probabilistic aggregation, ensemble learning

1 Introduction

Since the 1980s, the increasing sophistication of machine learning and computer technologies has enabled the development of solutions to a variety of challenges facing the Natural Language Processing (NLP) community. Knowledge discovery systems are of particular interest to commercial, industrial, and government organizations that rely on computer processing to perform transactions, evaluate consumer demands, and, in general, draw conclusions and make decisions that depend upon an extensive knowledge base. Often, construction of such a knowledge base depends upon the automatic extraction of relational information and, more fundamentally, related named entities[1] (e.g., people, organizations) from a collection, or *corpus*, of text documents (e.g., e-mail, news articles). Consequently, the reliability of these systems is highly susceptible to extraction errors.

However, even state-of-the-art entity extraction tools are vulnerable to variations in (1) the source and domain of a corpus and its adherence to conventional lexical, syntactical, and grammatical rules; (2) the availability and reliability of manually annotated data; and (3) the complexity of entity types targeted for extraction. Under these and other challenging conditions, extractors produce a range of interdependent errors that mangle entity output and rarely achieve adequate accuracy rates for practical use. However, many extraction technologies, distinguished by the nature of their underlying algorithms, possess complementary characteristics that may be combined to selectively amplify their attractive attributes (e.g., low miss or false alarm rates) and mitigate their respective weaknesses.

Many previous extractor combination methodologies that aim to leverage these characteristics rely upon variations of a "voting" mechanism (e.g., majority vote [1]). In practice, such approaches often fall short, as they depend heavily upon the number and type of extractors chosen, and they do not account for variations in the underlying extraction methodologies and the differing characteristics of their errors. Moreover, such systems tend to be limited in their ability to assess uncertainty, a critical capability for evaluating reliability in downstream analysis and decision-making. Proposed enhancements to the basic voting mechanism include, but are not limited to, weighting of constituent (i.e., *base*) extractors' output [2]; stacking of base extractors [3]-[5]; establishing a vote "threshold" [6]; and bagging [7]. Even more sophisticated techniques, such as that described in [8], fail to adequately account for the impact of text within a local neighborhood of a word of interest. A method based on the Conditional Random Field (CRF) model presented by Si, et al. [9], demonstrated that performance is enhanced by incorporating the classification structure of nearby words.

The aggregation methodology described in this paper, i.e., the entity *meta-extractor*, represents a significant departure from previous combination techniques. Specifically, the meta-extractor harnesses the unique characteristics of its base extractors via the estimation of conditional probability distributions over a space of extraction errors defined relative to the entities themselves. The resulting performance profiles are used to determine the most likely truth, in a probabilistic sense, given extractor output in a *region* of text.

Section 2 of this paper will describe the probabilistic characterization of base extractor performance, and Section 3 will focus on the construction and ranking of hypotheses. In Section 4, empirical results will be presented showing that the meta-extractor achieves statistically significant improvements

[1] We will often use "entity" and "named entity" interchangeably.

in entity extraction over its base extractors and a majority vote algorithm. Finally, we will discuss our conclusions and future research in Section 5.

2 Base extractor performance

In the remaining discussion, we assume that an entity can be expressed as a string (i.e., *name*) associated with a *location*[2] in the source text. To enable the characterization of base extractor performance, we assume an annotated set of documents is available (distinct from those used for training) to serve as an "evaluation corpus" for the base extractors[3]. The *ground truth* entity data, *G*, consists of the true (i.e., manually annotated) entities identified in the evaluation corpus. The meta-extractor aggregates the output of $K > 1$ base entity extractors, where D_k denotes the output of extractor *k* relative to a corpus. When the locations of a ground truth and extracted entity intersect, we say that the entities *overlap*[4].

2.1 Transformations of entity data

Entity extractors are generally of three basic types: rule-based, statistical and heuristic. Despite their algorithmic differences, however, their common objective is to correctly extract fragments from text that represent real-world entities, such as people, organizations, or locations. At a high level, this task may be regarded as a three-stage process in which an extractor (in some prescribed order) must *detect* a reference to an entity in a document, *identify* the offsets that delineate the name of a detected entity, and *classify* it as to its type. We focus chiefly on the first two of these in this paper.

Many of the most effective extractors are proprietary, and hence, direct analysis of their underlying algorithms is often infeasible. Therefore, we choose to treat each extractor *k* as a "black box". However, mistakes that are made on an annotated corpus result in observable discrepancies between its output, D_k, and the known ground truth, *G*. Thus, *G* serves as a baseline relative to which extractor behaviors can be characterized. More formally, the extraction process can be regarded as a *transformation* from *G* to D_k, denoted by $\tau(G,D_k)$, that is driven by the occurrence of extraction errors. Hence, assessing the performance of a base extractor lies in characterizing the types and propensity of the errors driving this transformation. Unfortunately, *G* and D_k can be very large, so it is prudent to decompose $\tau(G,D_k)$ into an ordered collection of smaller, more manageable (i.e., elementary) transformations; i.e., $\tau(G,D_k) \equiv \{\tau_i(G_i,D_{ki})\}$, where G_i and D_{ki} are subsets of *G* and D_k respectively.

Ground Truth:	*"Barack Obama"*	*"United States"*
Source Text:	**...President Barack Obama of the United States...**	
Extractor 1:	*"President Barack Obama"*	*"the United States"*
Extractor 2:	*"President Barack Obama of the United States"*	

Fig. 1. Ground truth entity data and corresponding extracted data.

The elementary transformations τ_i occasionally assume complex forms. In Fig. 1, for example, the output of Extractor 2 corresponds to a transformation of two ground truth entities into one extracted entity. Therefore, we choose to define the τ_i's in terms of the number of ground truth and extracted entities that they involve, subject to a desired set of properties. We now specify these properties more formally.

Let $\tau_i(G_i,D_{ki}) \equiv \tau^{mn}$ exactly when G_i is an ordered set of $m \geq 0$ consecutive ground truth entities and D_{ki} is an ordered set of $n \geq 0$ consecutive extracted entities from extractor *k*, where at least one of *m* and *n* is strictly positive[5]. The set of allowable types forms a *transformation space* given by $\mathbf{T} = \{\tau^{mn} : m,n \geq 0, m+n > 0\}$. For a set of elementary transformations $\{\tau_i(G_i,D_{ki})\}$ that comprise $\tau(G,D_k)$, we require that the following hold:

1) For all $g \in G$, there is exactly one $\tau_i(G_i,D_{ki})$ such that $g \in G_i$; similarly, for all $d \in D_k$, there is exactly one $\tau_j(G_j,D_{kj})$ such that $d \in D_{kj}$;
2) If $g \in G$ and $d \in D_k$ overlap, then there exists some $\tau_i(G_i,D_{ki})$ such that $g \in G_i$ and $d \in D_{ki}$;
3) Any $\tau_i(G_i,D_{ki})$ cannot be partitioned into two or more transformations satisfying both (1) and (2).

Under these properties, the entities extracted by Extractors 1 and 2 in Fig. 1 correspond to two $\tau^{1,1}$ transformations and one $\tau^{2,1}$ transformation, respectively.

It can be easily shown that properties (1)-(3) are necessary and sufficient to determine a unique collection of elementary transformations that partition $\tau(G,D_k)$, a desirable condition to ensure consistent meta-extractor performance. However, the space of $\tau^{m,n}$ transformation types is massive, and transformations become rarer as *m* and *n* become large. Hence, from a practical perspective, annotated data may be too sparse to compute reliable probability estimates over an unabridged transformation space. To that end, we relax property (1) above so that we can further decompose rare transformation types into a combination of simpler, overlapping transformation types that are more frequently observed. Care must be taken to ensure that the partition derived from a reduced space of transformation types is unique. We have typically limited the space to $\mathbf{T} = \{\tau^{0,1}, \tau^{1,0}, \tau^{1,1}, \tau^{1,2}, \tau^{2,1}\}$.

[2] We express the location of a text string in terms of its start and end offsets relative to the first character in the corpus.

[3] Three distinct corpora are required for: (1) training the base extractors, (2) evaluating their performance, (3) testing the meta-extractor.

[4] We generally assume that ground truth entities do not overlap and that the entities extracted by extractor *k* do not overlap.

[5] $\tau^{0,0}$ refers to the transformation involving no true or extracted entities at a corpus location. This event is not directly observable or easily characterized.

Many of these elementary transformations, e.g., $\tau^{m,n}, m \neq n$, encapsulate a variety of common extraction errors. For example, an extractor may detect one entity where there are, in fact, three. Since these types of errors are implicitly accounted for via the transformation space, we can think of these as *implicit* errors (which, notably, include the Miss and False Alarm errors, $\tau^{1,0}$ and τ^{01}, respectively). However, observe that the $\tau^{2,1}$ transformation in Fig. 1 contains additional discrepancies between the true and extracted entities that the transformation type does not embody. Specifically, the output of Extractor 2, "President Barack Obama of the United States", includes the extra text "President" and "of the". These and other discrepancies within instantiated transformations can be regarded as *explicit* errors and are mapped into a set of error types, $\mathbf{E} = \left\{ e_1, e_2, \ldots, e_s \right\}$, called an *error space.*

2.2 The error space

Though we place no specific constraints on the cardinality of the error space, the granularity of **E** must be considered. That is, a coarse error space may prevent subtle extractor behaviors from being adequately characterized, but an error space that is too fine may cause probability estimation to be problematic when annotated data are sparse.

To illustrate these concepts, suppose we define the space of discrepancies to consist of all possible ways that "extra characters" can corrupt an entity name. Then the three spaces defined in Eq. (1) each constitute a valid error space.

$$\begin{aligned}
&(\mathbf{E}_1) \quad e = \text{"extra characters"} \\
&(\mathbf{E}_2) \quad e_l = \text{"extra characters + name"}, \\
&\qquad\quad\; e_r = \text{"name + extra characters"} \qquad (1) \\
&(\mathbf{E}_3) \quad e_{l_i} = \text{"}i\text{ extra characters + name"},\ i = 1,2,\ldots,k \\
&\qquad\quad\; e_{r_i} = \text{"name + }i\text{ extra characters"},\ i = 1,2,\ldots,k
\end{aligned}$$

Observe that the respective cardinalities of $\mathbf{E}_i$ in Eq. (1) are given by $\left|\mathbf{E}_1\right| = 1$, $\left|\mathbf{E}_2\right| = 2$, and $\left|\mathbf{E}_3\right| = 2k$. In the empirical studies presented in Section 4, we have utilized an error space defined as in Eq. (2).

$$\begin{aligned}
e_x &= \text{"extra characters"}, \\
e_m &= \text{"missing characters"}
\end{aligned} \qquad (2)$$

Ultimately, the choice of an appropriate mapping (and hence, **E**) may be influenced by many factors that depend upon the application in question and its associated requirements. However, as mentioned above, the amount of annotated data available for estimating probability distributions over transformation and error types (i.e., implicit and explicit errors) will likely play a critical role.

2.3 Error probability estimation

For each base extractor *k*, we must estimate a probability distribution over a transformation space, **T**, and an error space, **E**. At a high level of abstraction, **T** and **E** are related hierarchically; that is, explicit errors occur within observed transformations, and it is natural to exploit this dependency. Specifically, we compute the relative frequency of each transformation type in the evaluation corpus, along with the relative frequency of each error *conditioned* on transformation type. In determining the latter, an explicit error of type $e_j \in \mathbf{E}$ may occur more than once in conjunction with a transformation (depending on **E** and **T**). However, we make the simplifying assumption that within an observed elementary transformation, explicit errors of different types may co-occur, but those of the same type may not.[6] Accordingly, we say the *state* of each explicit error is binary, and is given by

$$\begin{aligned}
s_{\tau_i}(e_j) &= \begin{cases} 1, \text{ if } e_j \text{ occurs within } \tau_i \\ 0, \text{ otherwise} \end{cases} \\
s_{\tau_i}(\mathbf{E}) &= \{s_{\tau_i}(e_j)\}_{e_j \in \mathbf{E}}, \tau_i \in T_k
\end{aligned} \qquad (3)$$

where $T_k = \{\tau_i(G_i, D_{ki})\}$ is the set of elementary transformations that form $\tau(G, D_k)$, and $s_{\tau_i}(\mathbf{E})$ is the joint state of all defined error types within $\tau_i(G_i, D_{ki})$.

To exemplify this concept, suppose we observe the $\tau^{1,1}$ transformation "Barack Obama" → "when Barack Obama was elected". No implicit errors are associated with this transformation type, but the set of explicit errors that occur relative to error space $\mathbf{E}_1$ in Eq. (1) is $\{e\}$. Similarly, for $\mathbf{E}_2$, $\{e_l, e_r\}$; and for $\mathbf{E}_3$, $\{e_{l_5}, e_{r_{12}}\}$. We can estimate the conditional probabilities of the explicit error types in **E** for extractor *k* via the expression

$$\hat{P}_k(e_j \mid \tau^{m,n}) = \frac{\sum\limits_{\tau_i \in T_k} s_{\tau_i}(e_j \mid \tau^{m,n})}{\sum\limits_{\tau_i \in T_k} I_{\tau_i}(\tau^{m,n})}, e_j \in \mathbf{E} \qquad (4)$$

where $I(\cdot)$ is the indicator function, $\tau^{mn} \in \mathbf{T}$ is a given transformation type, and $s_{\tau_i}(e_j \mid \tau^{mn})$ and $I_{\tau_i}(\tau^{m,n})$ are defined to be 0 if transformation τ_i is not of type τ^{mn}. Similarly, the probability estimate for each transformation type is given by

$$\hat{P}_k(\tau^{m,n}) = \frac{\sum\limits_{\tau_i \in T_k} I_{\tau_i}(\tau^{m,n})}{\left|T_k\right|}, \tau^{m,n} \in \mathbf{T}. \qquad (5)$$

Note that there are a variety of alternative estimates that one might propose. Those defined in Eq. (4) and Eq. (5) were chosen for their computational simplicity and because they provide reasonable estimates of the quantities of interest assuming modest amounts of data.

[6] In our empirical studies, we found that relaxing this assumption generally had negligible impact on meta-extractor performance.

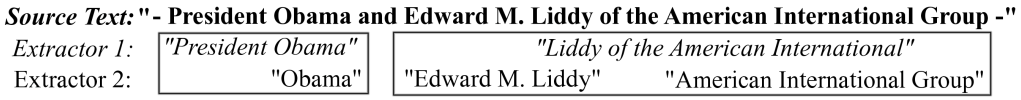
Source Text: "- President Obama and Edward M. Liddy of the American International Group -"
Extractor 1:
"President Obama"
"Liddy of the American International"
Extractor 2:
"Obama"
"Edward M. Liddy"
"American International Group"

Fig. 2. Two meta-entities are formed from overlapping extracted data – "President Obama", "Edward M. Liddy of the American International Group".

3 Extractor aggregation

In this section, we present a technique for aggregating base extractor output that leverages their performance characteristics to probabilistically rank hypothesized entities. This ranking forms the basis for determining meta-extractor output and associated confidence.

3.1 Meta-entities

In an operational setting, the base extractors are applied to a corpus for which ground truth is unknown. Using *only* the extracted output of its K base extractors[7], the meta-extractor must determine the truth, G.

Lacking access to the source text, the overlapping entities extracted by all of the base extractors at a given location in the corpus contain all the *available* information regarding the underlying ground truth at that location. The union of this overlapping extracted data yields a *meta-entity*, a novel construction used to establish a hypothesis space associated with this ground truth. Fig. 2 provides an excerpt of source text overlaid with the output of two hypothetical extractors, whose extracted data form two meta-entities.

3.2 The hypothesis space

We assume that any true entities must lie strictly within the corresponding meta-entity boundaries. Given this assumption, it only remains to determine the unique combination of words in the meta-entity that exactly matches these entities. To this end, we construct a hypothesis space that consists of all possible forms the ground truth entities may take. For example, the "President Obama" meta-entity in Fig. 2 yields a hypothesis space consisting of the following:

1) "President Obama"
2) "President", "Obama"
3) "President"
4) "Obama"
5) " " (i.e., the NULL hypothesis)

For small meta-entities it is feasible to generate the hypothesis space exhaustively. However, the space grows exponentially with meta-entity size and may be pared down by means of prior knowledge and/or assumptions. Such size constraints have not significantly impacted performance in our empirical studies.

Furthermore, the assumption that the ground truth entities lie entirely within the meta-entity boundaries does not always hold and may at first seem unreasonable. Indeed, when this assumption does not hold, the hypothesis space generated from the meta-entity will not contain the truth, and we say that the hypothesis space is not *closed*. In such cases, the meta-extractor will be unable to discover the truth.

Note, however, that the *closure rate*[8] of the hypothesis space is closely related to the number and diversity of the base extractors. In our empirical studies, utilizing four very different open source entity extractors, the truth was contained in the hypothesis space as often as 98% of the time. This finding suggests that for practical purposes, our assumption may, in fact, be reasonable. Strategies for increasing the closure rate include expanding the collection of base extractors, or enabling access to the source text during hypothesis space generation.

3.3 Ranking hypotheses

Given the hypothesis space Ω_x corresponding to a meta-entity x and the overlapping output D_{kx} of base extractor k, the *likelihood* of each hypothesis $H_{jx} \in \Omega_x$ must be computed. Under the assumption that H_{jx} is true, and provided the transformation and error spaces are appropriately defined, there is a unique set of transformations T_{jk} and associated explicit errors that transforms H_{jx} into D_{kx}. This is called the *error pathway* between the hypothesis and the extracted data. For example, let our hypothesis be H_j: "President", "Obama" in reference to meta-entity "President Obama" from Fig. 2. Based on the assumption that H_j is true, the pathway generated by Extractor 1 consists of a $\tau^{2,1}$ transformation with no explicit errors, whereas that generated by Extractor 2 consists of a $\tau^{1,1}$ and $\tau^{1,0}$ transformation with no explicit errors. Since each hypothesis induces a unique pathway, computing its likelihood reduces to estimating the probability of observing this pathway.

Hence, the likelihood of each hypothesis can be expressed as a function of the probabilities estimated as described in Section 2.3. Let $H_{jx} \in \Omega_x$ be the hypothesis of interest and $D_x = D_{1x} \cup D_{2x} \cup \ldots \cup D_{Kx}$ be the corresponding (i.e., overlapping) data extracted by the K base extractors. We estimate the conditional probability of H_{jx} given the observed extracted data D_x, via the following expression:

$$\begin{aligned} P\left(H_{jx} \middle| D_x\right) &\propto P\left(D_x \middle| H_{jx}\right) \cdot P\left(H_{jx}\right) \\ &= P\left(D_{1x}, D_{2x}, \ldots, D_{Kx} \middle| H_{jx}\right) \cdot P\left(H_{jx}\right), \end{aligned} \tag{7}$$

[7] To address efficiency requirements of certain real-world applications, we assume that the source text cannot directly be accessed.

[8] The *closure rate* is defined as the relative frequency with which hypothesis spaces contain the corresponding truth.

where $P(D_{1x}, D_{2x}, \ldots, D_{Kx} | H_{jx})$ is the joint conditional probability of the extracted data produced by the base extractors, and the prior probability of H_{jx} is given by $P(H_{jx})$. If desired, Eq. (7) can be simplified via various assumptions, such as assuming a uniform prior over $H_{jx} \in \Omega_x$ and/or statistical independence of the base extractors, transformations and errors. In many of our studies, we have assumed a uniform prior over the hypothesis space. Additionally, due to the data sparseness associated with many real-world applications, we have assumed independence of the extractors and transformations, as well as conditional independence of the explicit errors. Based on these assumptions, $P\left(H_{jx} \middle| D_x\right)$ can be expressed as follows:

$$P\left(H_{jx} \middle| D_x\right) \propto P\left(D_{1x}, D_{2x}, \ldots, D_{Kx} \middle| H_{jx}\right) = \prod_{k=1}^{K} \left(P\left(D_{kx} \middle| H_{jx}\right)\right) \quad (8)$$

where

$$P\left(D_{kx} \middle| H_{jx}\right) = \prod_{\tau_i \in T_{jk}} \sum_{\tau^{m,n} \in \mathbf{T}} P_k\left(s_{\tau_i}(\mathbf{E}) \mid \tau^{m,n}\right) P_k\left(\tau^{m,n}\right)$$

$$P_k\left(s_{\tau_i}(\mathbf{E}) \mid \tau^{m,n}\right) = \prod_{e_j \in \mathbf{E}} P_k\left(s_{\tau_i}\left(e_j\right) \mid \tau^{m,n}\right)$$

and $P_k(s_{\tau_i}(\mathbf{E}) \mid \tau^{m,n}) = 0$ if transformation τ_i is not of type $\tau^{m,n}$.

The null hypothesis, $H_{0x} = \varnothing$, is a special case and is handled slightly differently. Given that H_{0x} is true, the error pathway associated with the output of each base extractor will be composed of either $n > 0$ τ^{01} transformations or one $\tau^{0,0}$ transformation. Though we do not directly estimate $P_k(\tau^{0,0})$ for the base extractors, τ^{01} and $\tau^{0,0}$ are disjoint and are the only transformation types that can occur under this assumption. Hence, $\hat{P}_k(\tau^{0,0}) = 1 - \hat{P}_k(\tau^{0,1})$ constitutes a reasonable estimate.

Once each likelihood has been computed, the hypotheses can be ranked accordingly. In simple applications of the meta-extraction methodology the "winning" hypothesis may be accepted as the truth. However, the probabilistic ranking enables the quantification of uncertainty associated with the entity data. Moreover, it presents a framework for considering the top *n* competing hypotheses, or all hypotheses whose probabilities exceed a specified threshold. Effective strategies that exploit this ranking may yield significant rewards since, in our studies, the three highest ranked hypotheses contained the truth as often as 94.5% of the time. Ultimately, the choice of how to leverage the ranking depends upon the capabilities of the system utilizing this method and the requirements of the particular application domain.

4 Empirical studies

In this section, we present the results of two aggregation experiments using the output of (1) GATE, a rule-based extraction tool [10]; (2) LingPipe, an extraction tool based on Hidden Markov Models (HMMs) [11]; (3) Stanford Named Entity Recognizer (SNER), based on CRFs [12]; and (4) BALIE, an extraction tool that utilizes unsupervised learning [13]. These experiments were carried out using two publicly available annotated data sets: MUC 6 (Wall Street Journal) and CoNLL-2003 (Reuters).

The following studies focused upon two relevant real-world scenarios. The first involved a test in which the base extractors and the meta-extractor used identical training data. The meta-extractor, which requires annotated data for evaluation, used base extractors trained on less data, pitting weak learners against strong. To this end, MUC 6 was used in a 10-fold cross-validation procedure where, within each fold, 10% of the corpus was set aside for testing, and the remaining 90% was used to train and evaluate the base extractors.[9] The resulting ten performance estimates were bootstrapped (1000 samples) and displayed in a box plot (Fig. 3).

The second scenario assumes more challenging conditions in which the base extractors cannot be trained using representative data. These include cases where proprietary or rule-based extraction tools cannot be (re)trained and streaming text applications, where the source text is evolving over time and continual retraining of the base extractors is computationally infeasible. We simulated these conditions by training the base extractors on MUC 6 and then evaluating their performance and aggregating their output on CoNLL-2003. As in the first scenario, we performed 10-fold cross-validation, and the resulting estimates were bootstrapped and plotted (Fig. 3). For comparison, each plot includes performance estimates for a majority rule approach that is based on a simple B-I-O model [1].

4.1 Results

In Fig. 3, we have presented our results in terms of *F measure* (as computed in the CoNLL-2003 evaluation), Exact Match (EM) rates, and the combined Miss and False Alarm rates[10] for each base extractor, the majority vote algorithm, and the meta-extractor. We also assessed statistical significance in each case via a nonparametric pairwise test.

The top row in Fig. 3 presents the results generated for the first experimental scenario. The base extractors founded upon statistical methodologies, LingPipe and SNER, produced *F* measures that significantly exceeded those of GATE and BALIE. In general, we expected this behavior, since statistical methodologies often excel when they are trained on representative data. However, the performance of GATE clearly exceeded that of BALIE[11].

[9] Probability estimates were computed from the 90% via 9-fold cross-validation.

[10] These error types are often traded off to address operational requirements, but here we focus on the combined impact of both.

[11] BALIE was trained on a set of prepackaged untagged websites, negatively impacting its performance in our experiments.

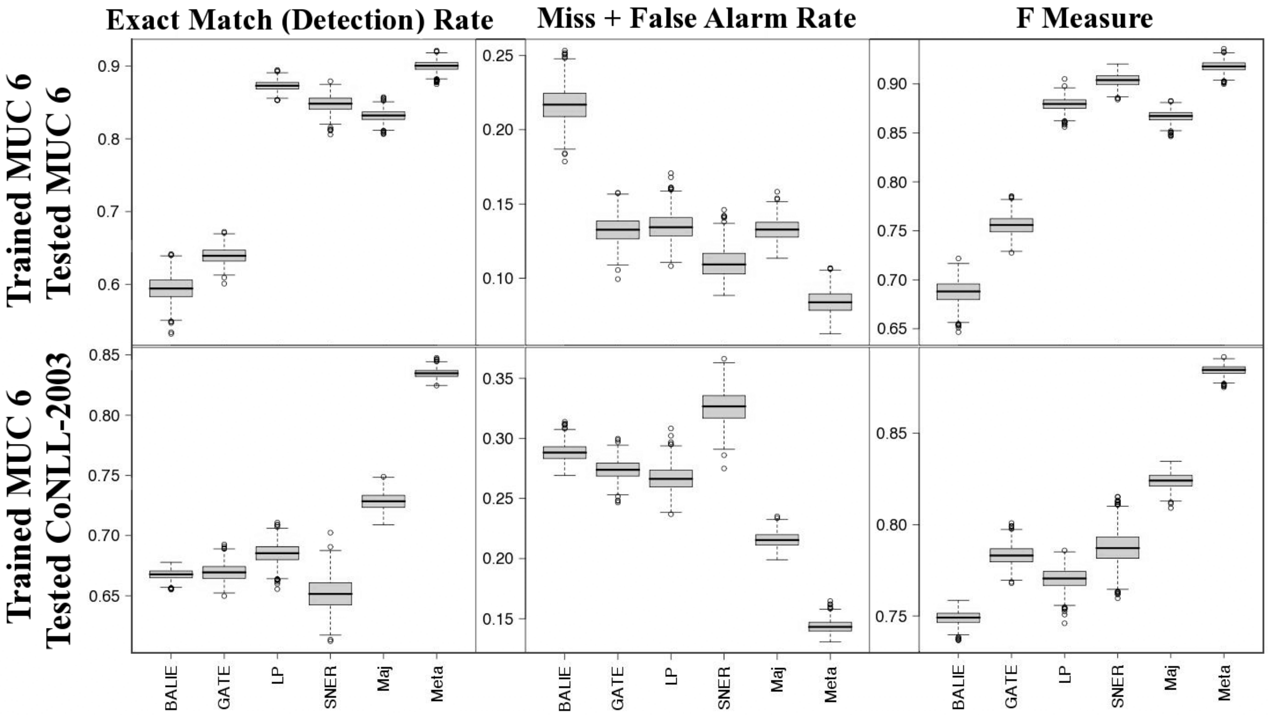

Fig. 3: Left to right: Exact Match, Miss+False Alarm, and F for the meta-extractor (Meta), majority vote (Maj) and the base extractors. The black line indicates the median; the box corresponds to the interquartile range (IQR); the whiskers extend to the most extreme point within 1.5 IQR of the median. The extractors were trained on MUC 6; the meta-extractor and majority vote were tested on MUC 6 (top), CoNLL-2003 (bottom).

Note that the improvement in the EM rate by the meta-extractor relative to LingPipe was significant, with a *p*-value of 0.002. For the other performance metrics, the improvement over the base extractors and majority vote was also highly significant, with $p = 0.001$. Unlike the majority vote algorithm, the meta-extractor improved upon both the high EM rate of LingPipe and the low miss and false alarm rates of SNER. This result illustrates the ability of the meta-extractor to leverage the respective strengths of all its base extractors, achieving improved overall performance.

In the second experimental scenario (Fig. 3, bottom), the degree of degradation in the performance of LingPipe and SNER was surprising. With respect to its EM rate and its miss and false alarm rates, SNER trails the other methods in this case. It is likely that the CoNLL-2003 source text differs considerably from MUC 6 with regard to its key characteristics. Also, the relative complexity of its underlying statistical model, the CRF, may render SNER more vulnerable to this scenario with regard to these metrics. In any case, GATE and BALIE appear to be fairly robust to these conditions.

Clearly, however, the meta-extractor was able to compensate for the failure of LingPipe and SNER. For all three performance metrics, the meta-extractor improvement over its base extractors was significant with a *p*-value of 0.001. With respect to majority vote, the naïve combination method achieved some success in the second scenario. However, in the former case, it was unable to effectively leverage the strengths of two strong extractors in the presence of two weak ones. This speaks to a severe limitation in such approaches. In contrast, the ability of the meta-extractor to directly leverage the strengths and weaknesses of its base extractors appears to provide it a considerable advantage.

4.2 Other considerations

In practical applications, standard metrics do not reflect the full range of advantages the meta-extractor provides. The construction of a hypothesis space that contains all possible forms of ground truth allows the meta-extractor to generate a ranking where the "winning" hypothesis is correct, even if the base extractors and the majority vote algorithm fail.

Table 1 presents an example of this phenomenon derived from the MUC 6 data set, in which all four base extractors

TABLE 1
RECONSTRUCTING THE TRUTH

Extractor	Extracted Entity 1	Extracted Entity 2
BALIE	"Federal Savings"	"Association"
GATE	"Valley Federal Savings"	"Loan Association"
LingPipe	"Valley"	"Federal Savings and Loan Association"
SNER	"Valley Federal Savings"	"Loan Association"
Meta	"Valley Federal Savings and Loan Association"	

Table 1: An example from MUC 6, where the meta-extractor reconstructed the truth. There were 233 hypotheses in the hypothesis space Ω. Majority voting fails in this instance.

incorrectly extracted portions of “Valley Federal Savings and Loan Association”. Naive voting methods favor the output of GATE and SNER, which are in complete agreement, and weighted voting methods might favor SNER, since it has been the most effective under ideal conditions. However, the meta-extractor correctly determined, based upon the performance profiles of its base extractors, that “Valley Federal Savings and Loan Association” was the most likely truth, with a probability of 0.333[12].

5 Conclusions

The experimental results presented provide evidence that the meta-extractor yields statistically significant improvements over its base extractors with respect to conventional summary metrics, exceeding the capabilities of a majority vote. In particular, it has demonstrated the ability to largely mitigate degradation due to operating conditions in which proper training of the base extractors is either computationally impractical or impossible. Moreover, we have observed that the constructed hypothesis space, when based on the output of the four extractors combined in this work, contains the truth as much as 98% of the time, and that the truth is contained in the top three ranked hypotheses as often as 94.5% of the time. This suggests that additional value may be achieved if the ranking can be exploited to its full potential.

Interestingly, the meta-extractor exhibits the ability to determine the underlying ground truth when all of its base extractors produce corrupted output. This capability provides obvious value to real-world applications, since highly corrupted entity data are a common occurrence when faced with the challenges associated with real data.

Important considerations in the application of this method to real-world problems motivated certain independence assumptions in the likelihood computation. Though the meta-extractor has successfully demonstrated that this aggregation methodology can be highly effective, we expect that, in general, these assumptions will seldom hold, and in some cases there may be a negative impact on meta-extractor performance. However, we conjecture that a joint probability model over the extractors, transformations and errors, though potentially more effective under data-rich conditions, would rapidly degrade when data are sparse. The simpler model may be more robust to these challenges and ultimately more practical in an operational setting. In light of these considerations, however, extending the meta-extractor to leverage joint information when sufficient annotated data are available may be justified.

Though these experiments used open source extractors, the meta-extraction framework possesses the capability to leverage any named entity extractor, proprietary or otherwise, whose performance can be quantified. Hence, new and more effective technologies developed for NLP can be readily incorporated. As such, this aggregation approach has the potential to provide long-term value in real-world entity extraction applications as it matures alongside the most effective technologies in named entity extraction.

Acknowledgements

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

6 References

[1] Z. Kozareva, O. Ferrández, et al., “Combining data-driven systems for improving named entity recognition,” *Data & Knowledge Engineering*, vol. 61-3, pp. 449-466, June 2007.

[2] D. Duong, B. Goertzel, et al., “Support vector machines to weight voters in a voting system of entity extractors,” in *Proc. IEEE World Congress on Computational Intelligence*, Vancouver, Canada, 2006, 1226-1230.

[3] H. Wang, and T. Zhao, “Identifying named entities in biomedical text based on stacked generalization,” in *Proc. 7th World Congress on Intelligent Control and Automation*, Chongqing, China, 2008, pp. 160-164.

[4] D. Wu, G. Ngai, and M. Carpuat, “A stacked, voted, stacked model for named entity recognition,” in *Proc. CoNLL-2003*, vol. 4, Edmonton, Canada, 2003, pp. 200-203.

[5] R. Florian, “Named entity recognition as a house of cards: classifier stacking,” in *Proc. 6th Conference on Natural Language Learning*, Taipei, Taiwan, vol. 20, 2002, pp.1-4.

[6] N. Kambhatla, “Minority vote: at-least-N voting improves recall for extracting relations,” in *Proc. COLING/ACL on Main Conference Poster Sessions*, Sydney, Australia, 2006, pp. 460-466.

[7] P. Kegelmeyer and M. Goldsby, “Massive ensembles for mindlessly improving named entity recognition,” unpublished.

[8] R. Florian, A. Ittycheriah, et al., “Named entity recognition through classifier combination,” in *Proc. CoNLL-2003*, vol. 4, Edmonton, Canada, 2003, pp. 168-171.

[9] L. Si, T. Kanungo, and X. Huang, “Boosting performance of bio-entity recognition by combining results from multiple systems,” in *Proc. 5th International Workshop on Bioinformatics*, Chicago, IL, 2005, pp. 76-83.

[10] H. Cunningham, D. Maynard, et al., “GATE: a framework and graphical development environment for robust NLP tools and applications,” in *Proc. 40th Anniversary Meeting of the Assoc. for Computational Linguistics*, Philadelphia, PA, 2002.

[11] Alias-I, LingPipe 3.8.2, 2008. http://alias-i.com/lingpipe.

[12] Stanford University, Stanford Named Entity Recognizer 1.1, 2008. http://nlp.stanford.edu/software/CRF-NER.shtml.

[13] University of Ottawa, Baseline Information Extraction (BALIE) 1.81, 2004. [Online] http://balie.sourceforge.net/.

[12] The second most likely hypothesis matched the output of GATE and SNER and had a probability of 0.214.